Version control with examples for SVN

Ben Trettel

University of Maryland, College Park

CFD group meeting
Monday, December 3, 2012

http://trettel.org/pubs/2012/Trettel-2012-SVN-presentation.pdf

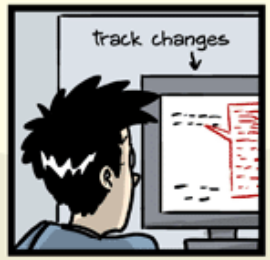
"FINAL".doc

FINAL.doc!

FINAL_rev.2.doc

FINAL_rev.6.COMMENTS.doc

FINAL_rev.8.comments5.
CORRECTIONS.doc



FINAL_rev.18.comments7.
corrections9.MORE.30.doc

FINAL_rev.22.comments49.
corrections.10.#@$%WHYDID
ICOMETOGRADSCHOOL????.doc

JORGE CHAM © 2012

What is a version control system?

A version control system is a way to manage changes to documents, source code, and other collections of information.

What do I use version control for?

- papers
- code
- notes
- C.V.

How version control helps

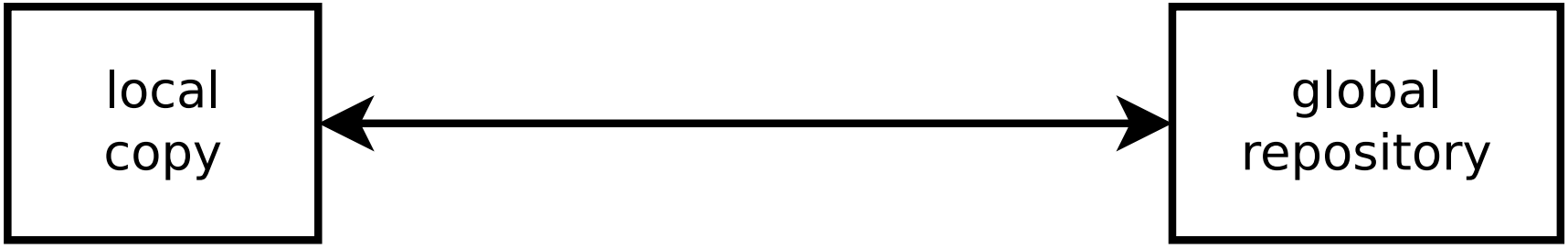
- Collaboration.
  - Once you commit a change, it's automatically added to everyone's code when they update.
- Tracking changes.
  - For posterity.
  - To see when you thought of something.
  - To help in debugging.
  - To roll back to an old version (to compare results, remove a bug that was added, etc.).
  - To add back features that were removed.
- Automatic backups.
- Encourages sharing of information and tools.

Terminology

- VCS — version control system
- DVC(S) — distributed version control (system)
- repository — a collection of information and its versions
- commit — tells the VCS that a file or files are ready to be shared

Distributed vs. centralized version control

Centralized version control



Distributed version control

Advantages and disadvantages of DVC

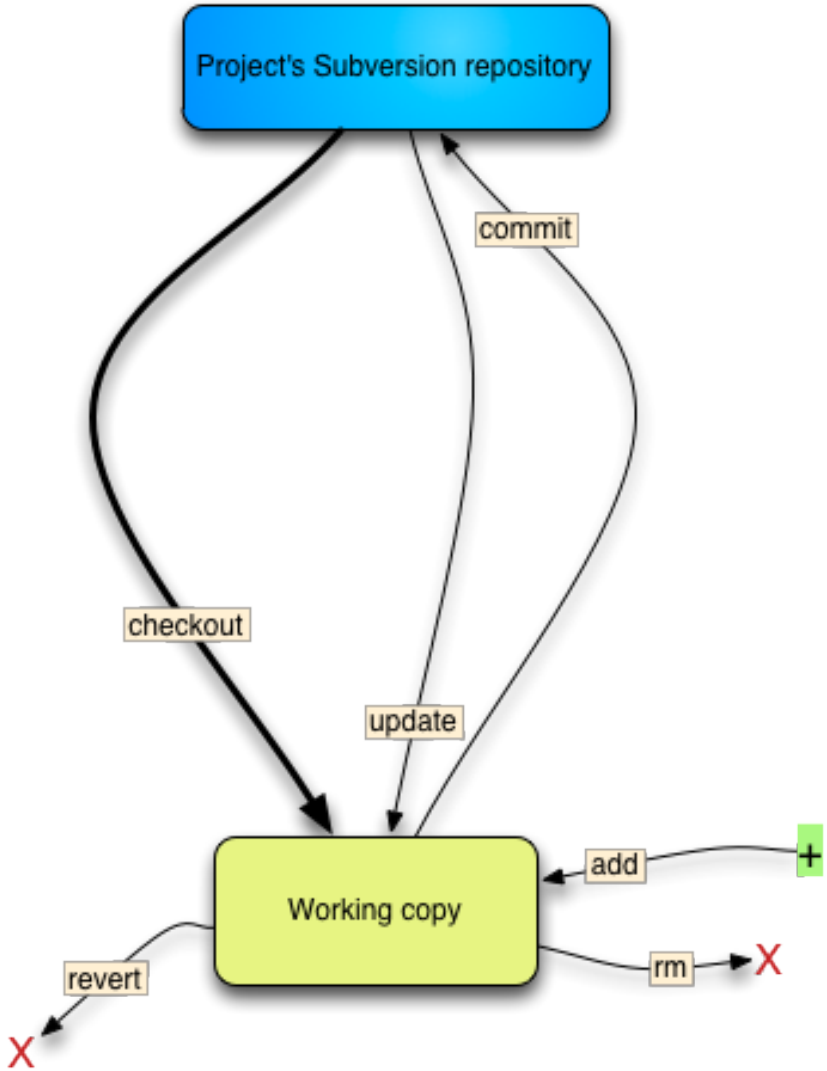
Advantages

- Many operations are fast because repository files are local.
- You always have a full backup of the repository.
- Can be easier to develop large changes you don't want to distribute to everyone yet.

Disadvantages

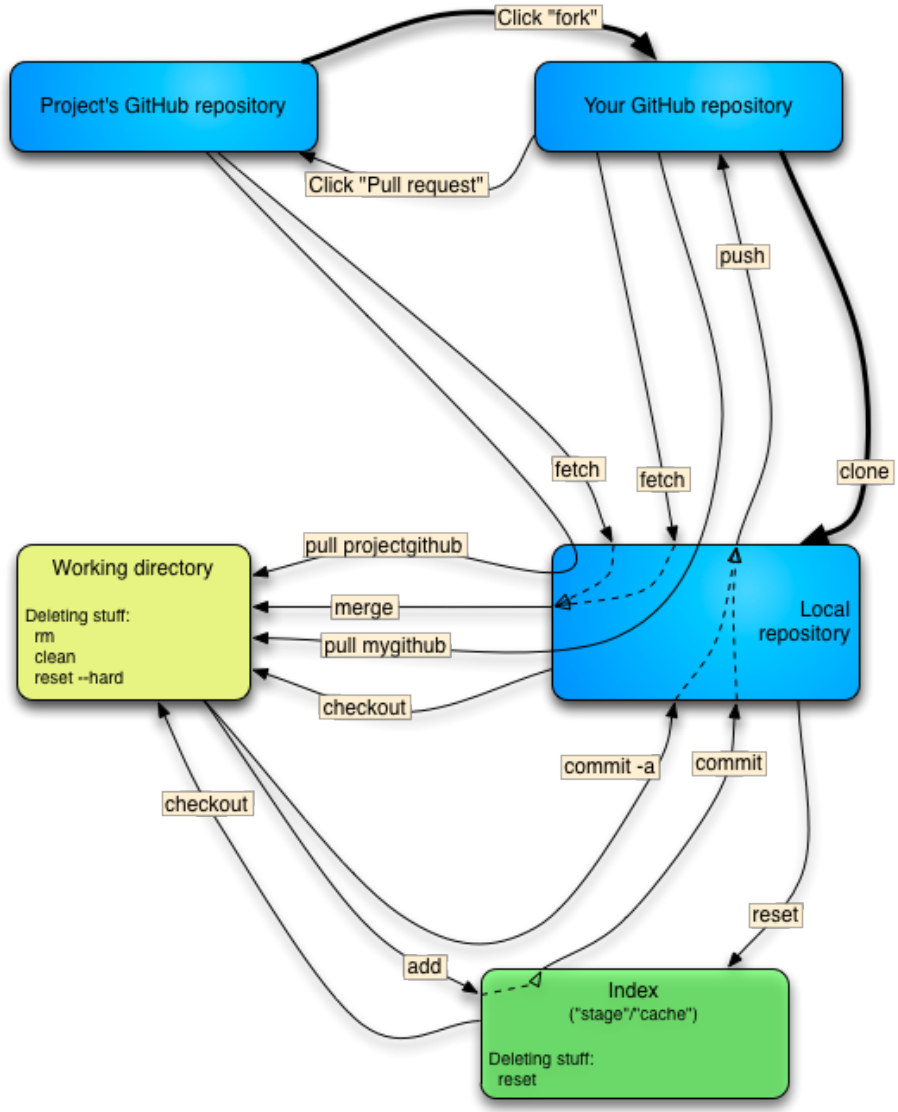
- Operations are more complex.

SVN (centralized)



https://git.wiki.kernel.org/index.php/GitSvnComparison

git (distributed)



https://git.wiki.kernel.org/index.php/GitSvnComparison

SVN vs. git

- SVN — popular centralized VCS
- git — popular distributed VCS

Advantages of SVN

- Simple and easy to learn. (Mercurial is an easier DVCS than git.)
- Robust tools and integration with other programs.
- Will add version information to files.
- Can have a global version number for a project.

Advantages of git

- Easier branching and merging. (Might be better for very large projects like the Linux kernel.)

Take home message

For moderate sized projects, it's more important that you use a VCS than that you use a specific VCS.

How to use SVN

- `svn checkout REPOSITORYURL` — get the latest copy of a project
- `svn update` — update the current directory and all below it to the latest
- `svn add FILENAME` — marks a file to be added in the next commit
- `svn delete FILENAME` — marks a file to be deleted in the next commit
- `svn copy SOURCE TARGET` — marks a file to be copied in the next commit (also see `move`)
- `svn revert SOURCE` — reverts changes made to a file (also undoes adds and deletions)
- `svn commit -m "Message text."` — commit changes in the current directory

http://www.abbeyworkshop.com/howto/misc/svn01/

http://www.addedbytes.com/cheat-sheets/download/subversion-cheat-sheet-v1.pdf

SVN keywords (Adding version data to files)

This is useful if you want a program (or even LaTeX document) to say which revision number it's at when you run.

1. `svn propset svn:keywords "Id Author Date Rev" FILE`
2. Now you can add the following lines to the file. These lines will be updated when you do `svn update`.
   - `$LastChangedBy:  $`
   - `$LastChangedRevision:  $`
   - `$LastChangedDate:  $`

   When updated you'll see something like this:
   - `$LastChangedBy:  ben $`
   - `$LastChangedRevision:  36 $`
   - `$LastChangedDate:  2012-10-04 22:10:37 -0400 (Thu, 04 Oct 2012) $`

SVN GUIs

- SmartSVN (Windows, Mac)
- RapidSVN (Linux)

.svn directory

Don't delete this. It's used by SVN to store information.

Good practices

- Test before you commit.
  - Compile in debug mode.
  - Run a related case to make sure it works correctly.
- Use descriptive commit messages. Putting keywords in the front helps when scanning through the logs.
  - e.g., *FDS User's Guide: Correct a name of a beta tester.*

Example: Getting the latest FDS source code

```
svn checkout
http://fds-smv.googlecode.com/svn/trunk/FDS/trunk/
fds-smv-read-only
```

https://code.google.com/p/fds-smv/wiki/Accessing_Subversion_Repository

Example: Committing a change

- command line
- GUI

Example: Putting version information in LaTeX

```
\usepackage{svn-multi}
\svnidlong
{$LastChangedBy: ben $}
{$LastChangedRevision: 36 $}
```

And at the end I have

```
Revision number: \svnrev
```

http://www.ctan.org/tex-archive/macros/latex/contrib/svn-multi

Setting up a new SVN project

1. `cd /home/svn/` (Create this directory if it does not exist.)
2. `sudo mkdir myproject`
3. `sudo svnadmin create /home/svn/myproject`
4. `sudo chown -R root:subversion myproject`[1]
5. `sudo chmod -R g+rws myproject`
6. `cd /home/svn/`
7. `svn co file:///home/svn/myproject`
   or if you are using a remote svn+ssh project
   `svn co svn+ssh://USERNAME@SERVER/home/svn/myproject`
8. `cd myproject`
9. `svn add *`
10. `svn commit -m "Added all myproject files." *`

[1]For Ubuntu, replace `root` with the username of a sudoer (usually you).

Some other useful commands and tools

- Graphical diff (meld, SmartSVN)
- Resolve a conflict
  `svn resolve FILENAME`
- Remove locks and complete operations.
  `svn cleanup`
- List all changed files.
  `svn status | grep -v ?`
- List all of a user's commits. *This is slow!*
  `svn log | sed -n '/username/,/-----$/ p'`
- Return to an older version.
  `svn update -r 1234`
- Edit existing log entry.
  `svn propset -r N --revprop svn:log "[new log message]" REPOSITORYURL --username USERNAME`